0

Mississippi State UNIVERSITY

Center for Air Sea Technology

1994 STUDENT RESEARCH PROJECTS

Technical Note 04-94
20 September 1994

Prepared for: Office of Naval Research (Code 1242)
Under Research Grant N00014-92-J-4109

Mississippi State University Center for Air Sea Technology
Stennis Space Center, MS 39529-6000

MISSISSIPPI STATE UNIVERSITY
CENTER FOR AIR SEA TECHNOLOGY

1994
STUDENT RESEARCH PROJECTS

BUILDING 1103-ROOM 233

STENNIS SPACE CENTER, MS 39529-6000

Compiled and Edited by
Lanny A. Yeske

This research was supported by the Department of the Navy, Office of the Chief of Naval Research, under Contract/Research Grant Number N00014-92-J-4109. The information contained in this publication does not necessarily reflect the position or the policy of the United States Government. No official endorsement should be inferred.

FOREWORD

The Center for Air Sea Technology (CAST) research program in FY94 was modified to adjust to new Navy Ocean Modeling and Prediction (NOMP) program priorities, especially in the area of coastal and semi-enclosed seas. The objectives were to:

- Conduct coastal and semi-enclosed seas ocean modeling basic research, embedded in a CAST modularized software system, with the emphasis on model relocatability to any geographical region;

- Support the technical requirements of Navy and university ocean modeling efforts by providing routine day-to-day technical support to the scientific staff, and by designing, developing, and implementing next generation technical support capability;

- Tailor and transition applicable advanced technical support capabilities developed for research community to the operational Navy; and

- Strengthen collaboration with academia by incorporating student and faculty in CAST projects.

In accomplishing these objectives, CAST in 1994 supported 11 graduate and undergraduate students, which included two through the MSU Cooperative Education Program, three from the MSU Department of Computer Science, one from the MSU NSF-sponsored Engineering Research Center, two through the University of Southern Mississippi Cooperative Education Program, and one each from Oregon State University, Tulane University, and Brandeis University. CAST also had a faculty program with four research affiliates from the MSU and Tulane University Departments of Computer Science, as well as the MSU-NSF Engineering Research Center.

This technical note summarizes the 1994 research conducted by these students and research affiliates. CAST was extremely pleased with the research support provided by these individuals, not only in their dedication but in the quality of the research conducted.

J.H. CORBIN
CAST Director

TABLE OF CONTENTS

Patrick Perrin
Ph.D. Program, Department of Computer Sciences
Tulane University

Title: Design of an Intelligent Support System for Scientific Databases

Objective: Scientific databases constantly and rapidly grow in terms of the amount of gathered knowledge. This huge flood of raw knowledge is mainly numerical data representing observations (e.g., satellite infrared images, ship and storm tracks, etc.). It is predicted to be of the order of gigabit or terabit of new raw data per day. These fast growing scientific databases render impossible, even unfeasible, human raw data analysis; hence the urgent need for automatic data and efficient human-machine interface. This project responds then to the need to find knowledge in the flood of data and to efficiently allow the user to access and use this knowledge. It is then concerned in filling the gap between data generation and data understanding for efficient and effective data exploitation, such as retrieval of information and inferences on current knowledge.

Approach: The approach is for the system to discover, in an automatic manner, relationships among the objects contained, or to be inserted, in the databases. This corresponds to finding interesting and useful patterns in the raw observational data stored in the databases. Such patterns represent an effective and efficient description of the data. Transforming those patterns into production rules to be inserted in the knowledge base of an expert system, and adding the facilities and power of an interactive conversational natural language interface, will allow the system to do efficient exploitation of large scientific databases. Designing and implementing such an Intelligent Support System (ISS) is the final goal in the research. The overall ISS functional components are depicted in Figure 1.

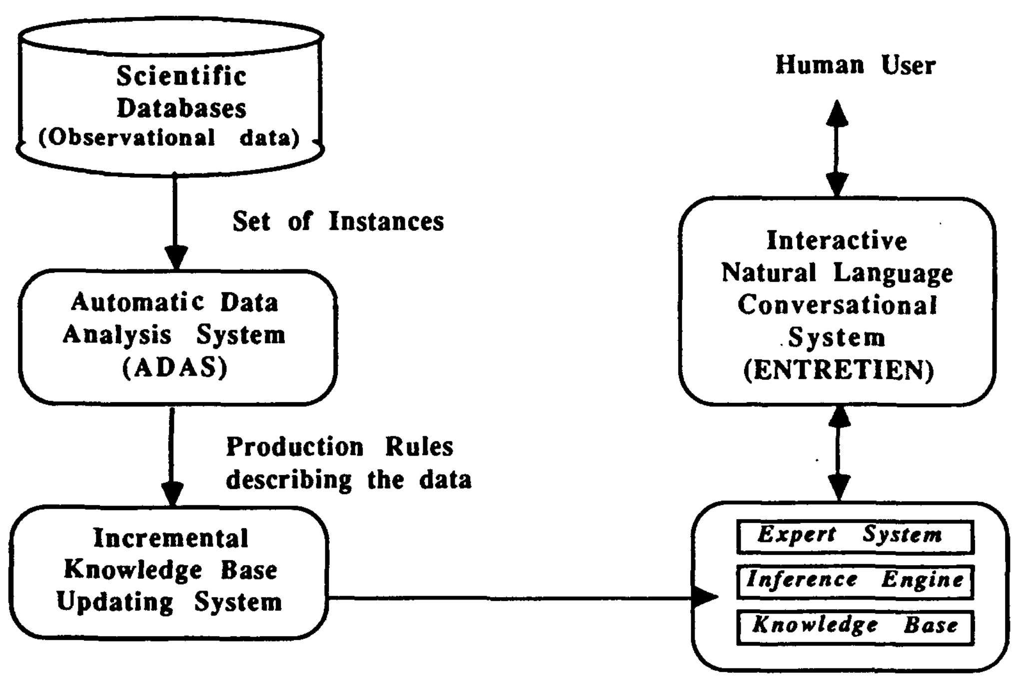

Figure 1. ISS Functional Components

This research is also concerned with the discovery-learning process of ISS. In developing an Automatic Data Analysis System (ADAS), which combines the strengths of two machine learning techniques, that are data clustering and induction learning by decision trees, the intent is to generate production rules that describe the raw data contained or being included in the scientific database. It continues and enhances previous work in which a system for knowledge discovery in scientific databases was designed and implemented using data clustering, classification heuristics, and domain knowledge. This previous system analyzed the raw data to find structure. This was done by partitioning the objects space into a hierarchy of clusters, each representing a class of objects considered to describe a similar concept. The clustering was performed using some pre-defined classification criteria (such as object features), some heuristics, and some domain knowledge. This research adds to this basic system a set of algorithms to make a general description of each cluster. Such descriptions are expressed in first order predicate logic as production rules. These production rules are intended to be used by an expert system for efficient exploitation of the knowledge contained in scientific databases (e.g., inference and information retrieval). These algorithms are based on inductive learning by decision trees and automatic rule generation.

The research also involved the effective exploitation of the knowledge by the user. Discoveries are appropriately formatted by the system for the intended user. In the first stage, this is done as first order predicate logic rules obtained from decision trees. This is a convenient knowledge representation for knowledge manipulation and analysis, and is easily transformable into natural language for human user data exploitation. We designed an interactive and robust natural language conversational simulation model called ENTRETIEN. Since language is a natural support for human being communication for transfer of information, ENTRETIEN will allow the human user to naturally communicate with the system to efficiently exploit the potential information non-trivially stored in the large scientific databases. This is to palliate the actual obvious lack of efficient and effective communications between human users seeking information and computer systems willing to provide such knowledge. ENTRETIEN will conduct the dialogue with the user by analyzing each user's utterance (e.g., query, command, etc.) expressed in a natural way (e.g., English) and in the current context of the conversation to better serve the user's needs. This is based on human natural language information seeking dialogues during which each participant attempts to recognize the other participant intention of plans for successfully communicating information. This system will thus interactively conduct the dialogue to recognize the user's intentions in seeking information in the scientific numerical databases. We want to base our plan recognition simulation model on a previously developed model.

Results: A series of experiments were conducted in applying ADAS to CAST-NEONS, a real world scientific database currently used by CAST for ocean modeling and prediction. By these experiments, it was shown that the system was able to identify automatically, from scratch, the Gulf Stream in the North Atlantic area by analyzing and classifying measures of its features, such as salinity and temperature. This was done at constant depth but we intend to enable the system to search more interesting correlations such as salinity, temperature, depth, and temporal variables. By these examples, we empirically demonstrate the feasibility of our approach in doing automatic numerical data analysis.

This research clearly shows that combining machine learning techniques (data clustering and induction learning by decision trees) with production rules is an effective mean for automatic data analysis of huge and rapidly changing amounts of observational raw data in scientific databases. In terms of expert systems, automatic rule generation from non-trivial knowledge contained in scientific databases eases the knowledge acquisition bottleneck that usually exists in extracting knowledge from human experts. Further developments of the prototype seems promising in solving the problem of automatic data analysis for information retrieval in scientific databases.

We believe that constructing the natural language interface, in which the computer "speaks" the same language as the human user, is of great interest to efficiently exploit information non-trivially stored in databases. We want to go a step further in building an intelligent machine capable of communicating efficiently, as humans do naturally.

Future Research: Future directions toward the goal of designing and implementing the intelligent support system include the design of the knowledge base updating systems, the integration of the productions rules and a previous semantic network that captures the domain knowledge into an expert system knowledge base, the implementation of ENTRETIEN for the system to naturally converse with human users, the selection of better data clustering algorithms to refine and render more robust the knowledge discovery/learning process, and finally to encode the knowledge found into a generic form suitable for the knowledge base.

Research Advisor: Dr. Frederick Petry, Department of Computer Science, Tulane University

Dongmei Wu
M.S. Program, Department of Computer Science
Mississippi State University

Title: Error Pattern Identification and Clustering (EPIC)

Objective: Scientists at the Naval Oceanographic Office (NAVOCEANO) at the Stennis Space Center use a numerical model developed by an international group to provide predictions of wave height for the Navy. The model which is called the Wave Model (WAM) predicates significant wave height (SWH) based on wind speed. Comparisons of the WAM predictions and SWH calculated from altimetry measurements have indicated that the WAM predictions are inaccurate in some situations. Because the altimetry SWH data cannot be obtained quickly enough and because it covers only a very narrow geographical region, the NAVOCEANO staff cannot use this data to correct the WAM predictions. Therefore research on using machine learning techniques to determine the situations in which the errors of the WAM predictions occur has been initiated. The objectives of this project are:

- in the short term, we want to identify features which characterize the circumstances in which errors in the Wave Model (WAM) output typically occur;
- in the long-term, we plan to investigate the use of techniques which can automatically exam historical data to identify the sources of errors, and to associate a degree of confidence with WAM data, and correct WAM output. We intend to develop a general approach which will be applicable to a variety of oceanographic model validation tasks.

Approach: Two well-known clustering packages, Cobweb/3 and AutoClass, have been used for this project. Cobweb/3 employs an incremental concept formation strategy for clustering and classification. It first forms a tree of one class which contains general concepts for observations, and then discriminates new observations from the others *in an incremental fashion*. The result is a classification hierarchy. AutoClass, on the other hand, uses the standard Bayes' rule to cluster observations into classes. AutoClass does not build a hierarchy.

NAVOCEANO has supplied one-year of data of WAM predictions and ERS-1 and TOPEX altimetry SWH data for the Mediterranean Sea. The altimetry data includes values of different features such as longitude, latitude, time, wind speed, and altimetry SWH. Values for bathymetry were obtained from the Center for Air Sea Technology. It was necessary to extract WAM predictions that corresponded to altimetry measurements in time and location. After the WAM data points had been extracted, the difference (SWH error) of the WAM prediction and the altimetry SWH were calculated using the following formula (based on the assumption that the altimetry SWH is accurate):

$$SWH_e = \text{altimetry SWH} - \text{WAM prediction}$$

For the experiments described in this report, data from ERS-1 for a thirty-day period has been used (every 10th data point was selected).

The two clustering algorithms have been applied to different sets of features:

- longitude, latitude, wind speed, altimetry SWH, SWH error, and bathymetry.
- longitude, latitude, wind speed, and SWH error.
- longitude, latitude, and SWH error.
- wind speed, altimetry SWH, and bathymetry with SWH error respectively.

- wind speed, altimetry SWH, SWH error, and bathymetry.

The experiments have been performed using two modes of the clustering algorithms: clustering and prediction. Data has been run in the clustering mode to see whether meaningful clusters can be obtained with different features. The prediction mode is used to determine the meaningfulness and effectiveness of the clusters for predicting values of certain attributes. In the prediction mode of Cobweb/3, a set of training data is used to form a classification hierarchy, and then a set of test data with a missing attribute is classified based on the classification hierarchy. In these experiments, the values of SWH error were predicted based on the other attributes. Training data and test data were different samples of ERS-1 data for the same time period.

For both clustering and prediction modes, the experiments have been done with both real values and nominal values since nominal values appeared to be easier to analyze. The nominal values, such as "low" and "deep", are actually tags for real values in specified ranges. For the current data, the real values of each attribute have been converted to nominal values using pre-defined ranges which have been approved by the NAVOCEANO scientists (Table 1).

Table 1. Ranges Used for Nominal Value

Attribute	Value Range	Nominal Value
Longitude	$-6.0 < x \leq 0$	VeryLow
	$0 < x \leq 10.0$	Low
	$10.0 < x \leq 20.0$	Medium
	$20.0 < x \leq 30.0$	High
	$x > 30.0$	VeryHigh
Latitude	$30 < x \leq 34$	VeryLow
	$34 < x \leq 38$	Low
	$38 < x \leq 42$	Medium
	$42 < x \leq 46$	High
	$x > 46$	VeryHigh
SWH	$0.0 \leq x < 1.5$	Low
	$1.5 \leq x < 3.0$	Medium
	$3.0 \leq x < 4.5$	MediumHigh
	$4.5 \leq x < 6.0$	High
	$6.0 \leq x < 7.5$	Higher
Wind Speed	$0.0 \leq x < 4.0$	VeryLow
	$4.0 \leq x < 8.0$	Low
	$8.0 \leq x < 12.0$	Medium
	$12.0 \leq x < 16.0$	High
	$x \geq 16.0$	VeryHigh

Table 1. Ranges Used for Nominal Value (continued)

Attribute	Value Range	Nominal Value
Bathymetry	$x < 10$	VeryShallow
	$10.0 \leq x < 50.0$	Shallow
	$50.0 \leq x < 300$	Medium
	$300 \leq x < 1000$	Deep
	$x \geq 1000$	VeryDeep
SWH error	$x < -2.5$	WAMVeryHigh
	$-2.5 \leq x < -1.0$	WAMHigh
	$-1.0 \leq x < 0.0$	WAMJustHigh
	$0.0 \leq x < 1.0$	WAMJustLow
	$1.0 \leq x < 3.0$	WAMLow
	$x \geq 3.0$	WAMVeryLow

We have developed a new method for analyzing the output clusters from experiments that use real values for attributes. This method uses a form of inferential statistics called the confidence interval of the mean. The confidence interval is a range of values bounded by a lower and an upper limit. This interval is expected to contain the mean of the parameter with a certain degree of confidence. In the current experiments, the degree of confidence is 95%. The formula used to compute the confidence interval is:

$$\text{Mean} \pm 1.96 \times \frac{\text{Standard-Deviation}}{\sqrt{\text{Object-Count}}}$$

For the prediction mode, the absolute value of the differences of the predicted SWH errors from Cobweb/3 and the original SWH errors were obtained. In order to analyze the differences, different ranges [0.0, 0.5], (0.5, 1.0], (1.0, 1.5], (1.5, 2.0], (2.0, 3.0], and (3.0, ...) have been defined, and the percentage of the difference values that occur in each range has been calculated. For example, if the range [0.0, 0.5] is an acceptable error range, and the calculated percentage for this range is high, then it can be said that the prediction is good. This allows us to compare the effectiveness of clusters that are based on different features in predicting the SWH error.

Results: We have observed that if longitude and latitude are used as attributes in the clustering experiments, the objects are clustered almost exclusively based on location.

We have observed that when compared to the ERS-1 data, WAM predictions are low in almost all locations. The experiments have shown that when the wind speeds increase, WAM predictions become even lower. Table 2 shows the clusters formed based on wind speed and SWH error. Our the experiments have not shown that the altimetry SWH and bathymetry have strong relationships with the SWH error. The use of all four attributes (wind speed, altimetry SWH, SWH error, and the bathymetry) did not seem to yield more meaningful clusters.

Table 2. Clusters Formed Based on Wind Speed and SWH Error

Cluster Number	Mean Wind Speed (m/s)	Mean SWH Error (m)
1	13.60	1.8
2	9.40	1.5
3	6.75	0.7
4	4.10	0.5
5	1.63	0.6

When we examined the clusters formed using nominal data, we found that the clusters were not as meaningful as those based on real values. We speculate that this is because some information is lost in the process of converting the real values to the nominal values.

For the experiments run in the prediction mode, more than half of the predictions of SWH error for each experiment are within the range of [0.0, 0.5] (Table 3).

Table 3. Analysis for SWH Error predictions for the Prediction Mode

Attributes used in clustering	Percent of Predicted Values in Each Range					
	[0.0, 0.5]	(0.5,1.0]	(1.0,1.5]	(1.5, 2.0]	(2.0, 3.0]	(3.0, ...)
Wind Speed SWH Error	58%	25%	8%	5%	1%	3%
Bathymetry SWH Error	38%	21%	14%	6%	11%	10%
Altimetry SWH SWH Error	53%	28%	10%	5%	3%	1%
Wind Speed Altimetry SWH Bathymetry SWH Error	54%	25%	8%	5%	5%	3%

Future Research: Our future research will be concentrated on determining other sets of features which may influence the SWH error. We will continue research on our long-term objective which is to investigate techniques which can automatically exam historical data to identify the source of errors, associate a degree of confidence with data, and correct WAM output.

Research Advisor: Dr. Susan Bridges, MSU Department of Computer Science

Zhifan Zhu, Ph.D. Program, Engineering Research Center
Mississippi State University

Title: Scientific Visualization of Oceanic and Meteorological Data

Objective: The study and prediction of weather changes in a global or mesoscale range requires knowledge of both atmosphere and ocean feature dynamics. Numerical weather and ocean models have been operational for many years, and coupled air-sea models have also appeared recently. In parallel, computing power has dramatically increased which allows higher time-space resolution modelling with feature-resolving capability. The potential data quantity exceeds hundreds of megabytes or even gigabytes. Visualizing features of interest is essential. The primary purpose of this research was to investigate the visualization techniques that must be efficient in automatically extracting and then visualizing important features from digital data. The work is considered as an extension of previous research. It includes improving the existing ocean feature detection algorithm and studying the meteorology feature detection methods.

Approach: To meet the research requirement, a new visualization environment/tool, envis, has been developed from a previous tool, vis3. The internal rendering coordinate system has been modified to allow visualization of either ocean data or atmospheric data, or both of them at the same time. The internal data buffer has been modified to be a five dimensional array to support multivariate time-varying volumetric data. The feature detection algorithms are again incorporated into the visualization system as supporting elements.

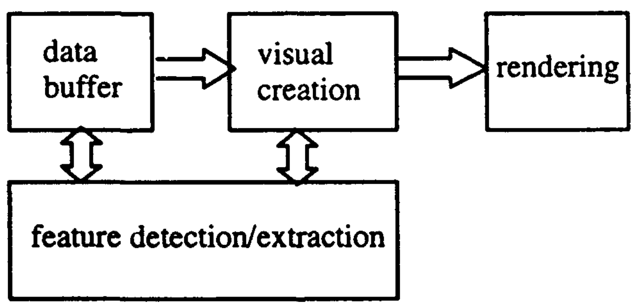

Figure 1. System Diagram

A new method has been used to detect ocean eddies. The previous approach of eddy detection is based on edge amplitudes and spatial correlations. The new algorithm works with the phases of edges. It first locates the interior points of a possible eddy by using a `gravity rule', which can be illustrated with an example of phase pattern (Figure 2). For a closed warm eddy, the phase angles along its boundary all point towards the center of the eddy. Therefore, any interior point of the eddy can not be moved out of the boundary without a sufficient force to overcome the center-bound gravity. For a cold eddy, the algorithm works in the same way except that all phase angles need be changed by 1805. This method is more reliable in detecting the eddy centers.

An equal amount of work has been paid to the study of feature detection for meteorological data. As a case study, a public-domain data set from Vis5d was used as the test data. The data set has 10 variables, 19 time-steps in a two-hour interval, a 35x41x15 grid with longitude resolution of 1.6 degrees, latitude resolution of 1.25 degrees, and about 1 km vertical resolution.

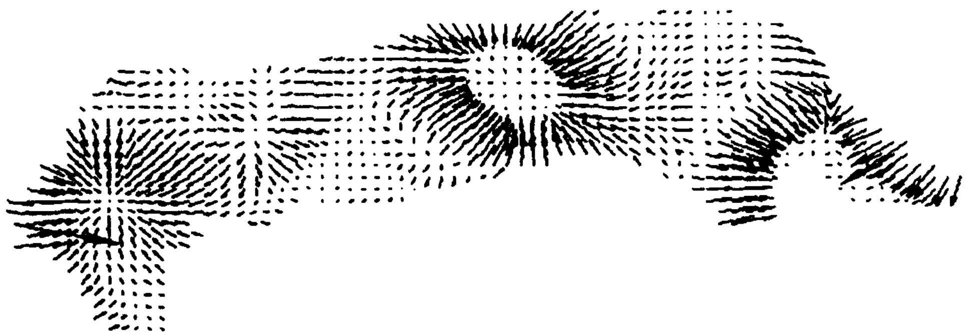

Figure 2. Phase Pattern Example

The effort was focused on cyclones and the jetstream at high altitude. The cyclone is a closed circulation about a low pressure center, which is counterclockwise in the Northern Hemisphere. The importance of understanding the cyclonic system has already been recognized in predicting and tracking thunderstorms. Such a meteorological feature is in fact associated with multiple atmospheric variables, such as wind, pressure. The algorithm that detects cyclones (anti-cyclones) involves computing the wind flow patterns and the vertical vorticity field.

The jetstream is relatively strong winds concentrated within a narrow stream in the atmosphere. In North America, the jetstream is of interest at mostly higher altitudes (above 8 km from the sea surface). It illustrates the massive circulation pattern that governs general air motion.

Results: Figure 3 shows an example of eddy detection with the new phase approach. The data is Gulf of Mexico temperature. The horizontal slice is clipped by the bathymetry (dark part). Two eddies are detected.

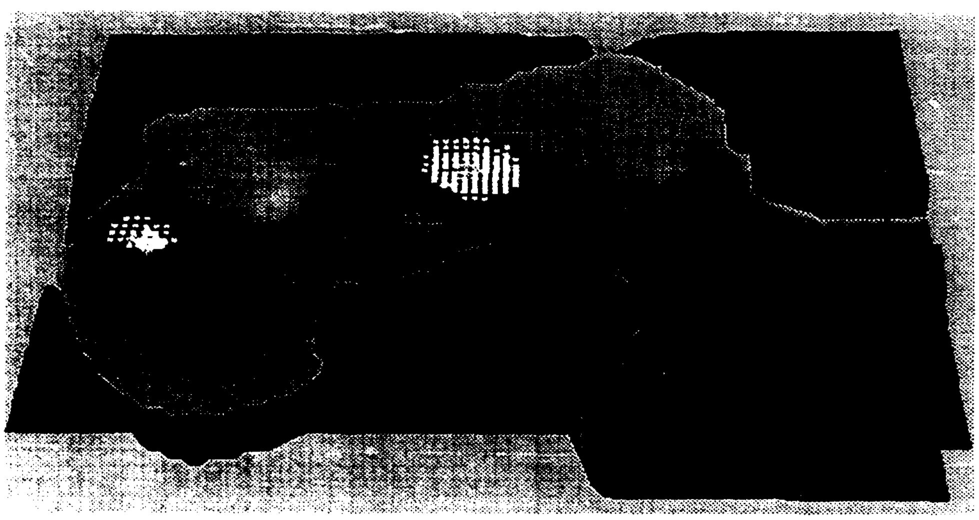

Figure 3. Two Detected Eddies

Figure 4.a. displays two detected cyclones. The bottom is the topography of North America. The bigger cyclone is around the central part of the United States, and the smaller one is at the ocean close to Alaska. The cyclone cores are represented by the vorticity surface. The white lines are the wind trajectories around the cyclone cores. They produce very vivid cyclone circulation patterns. The jetstream is shown by the strip with the arrow at the right end indicating the direction.

Figure 4.b. shows a detected cyclone, the jetstream and surface pressure contours. Again, the lines around the cyclone core are the wind trajectories, and the vertical vorticities are mapped. As can be seen, a low surface pressure center is perfectly matched with the detected cyclone in space.

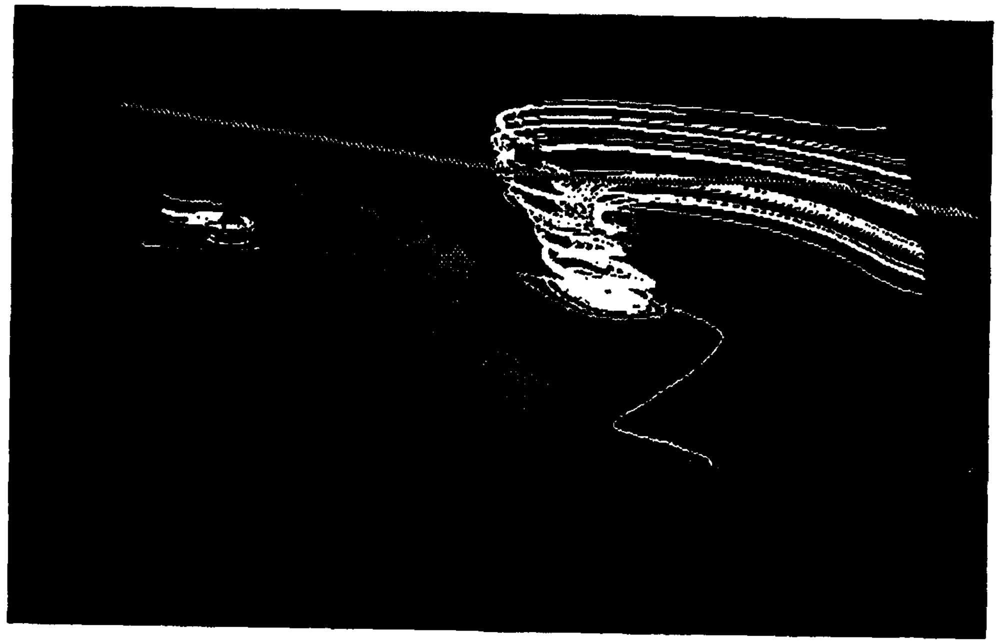

Figure 4a. Two Detected Cyclones

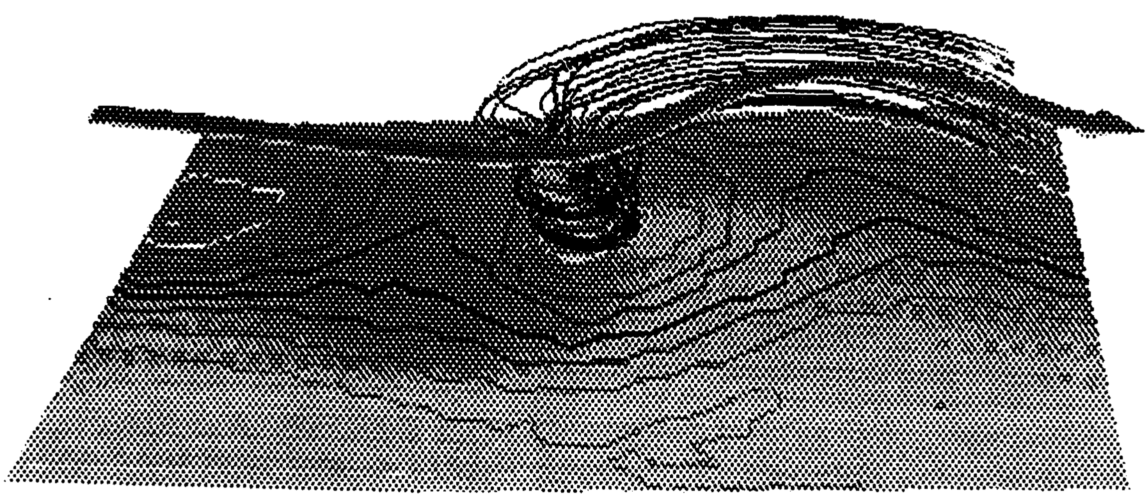

Figure 4b. One Detected Cyclone and Surface Pressure Distribution.

Future Research: Future efforts may include ocean eddy detection from current fields by using the similar techniques for cyclone detection, efficient recognition and tracking of feature movement along time, and visualizing features systematically.

Research Advisor: Dr. Robert J. Moorhead, NFS Engineering Research Center, Mississippi State University.

An Object–Oriented Prototype for a Geophysical Database
Subtask: Prototype Implementation

Chandrashekar Ramanathan
Julia Hodges
P.O. Drawer CS
Mississippi State, MS 39762
{shekar,hodges}@cs.msstate.edu

TITLE: Prototype Implementation of an Object–Oriented Geophysical Database

OBJECTIVE: The objective is to develop a prototype geophysical database by using a commercial object–oriented database management system (OODBMS). The data in the Naval Environmental Operational Nowcasting System (NEONS) relational database should be transformed and stored in the new system called ObjNEONS. It is envisaged that the resulting database system and the applications developed on top of the system will be more "natural" for the users and application developers by virtue of its object–orientedness.

APPROACH: Development of any database system involves at least three things: the data, a database schema for the data, and a DBMS. The NEONS database system served as a source of data for ObjNEONS. The database schema for this system is based primarily on an earlier work involved in providing an object–oriented *view* of the existing NEONS relational database. Noting that the objective is to develop an object–oriented database, a commercial OODBMS ObjectStore was chosen after doing a feature evaluation of the the various available systems.

Re–engineering an existing relational database system such as NEONS to an object–oriented version involves mapping the relational schema into an object–oriented (OO) schema. The OO schema should subsume the relational schema in such a way that there is no loss of information from the data. NEONS primarily consists of four types of data: image data, grid data, line data, and llt (lat–lon–time) data. The grid data type was chosen as the starting point for this implementation. The relational schema for grid data was transformed into an OO schema. This schema was then transformed into ObjectStore class definitions. These object classes were then instantiated using the data present in the NEONS database by yet another transformation of the relational tuples into objects and relationships. Figure 1 summarizes the whole process.

There are some similarities and differences between the current work and the earlier work of providing an object–oriented view of NEONS. Both of them involve schema and data transformations. However, in the current system, the transformed objects are persistent in the database that is managed by an OODBMS whereas in the earlier system, only the relational data was persistent and the objects were transient. Another point to note is that in the current system, the directions of the arrows indicating transformations in Figure 1 are uni–directional. That is, there is no provision for translating the objects into tuples that can be stored in the relational database. ObjNEONS may contain data not only ported from NEONS but also those that are newly generated by the scientists.

As mentioned earlier, the current work focuses on the grid data. A partial object–oriented schema for the grid data is shown in Figure 2. The notation provided by the FUSION object–oriented software development method is used in the object class diagram. The rectangular boxes represent object classes and the diamonds represent relationships.

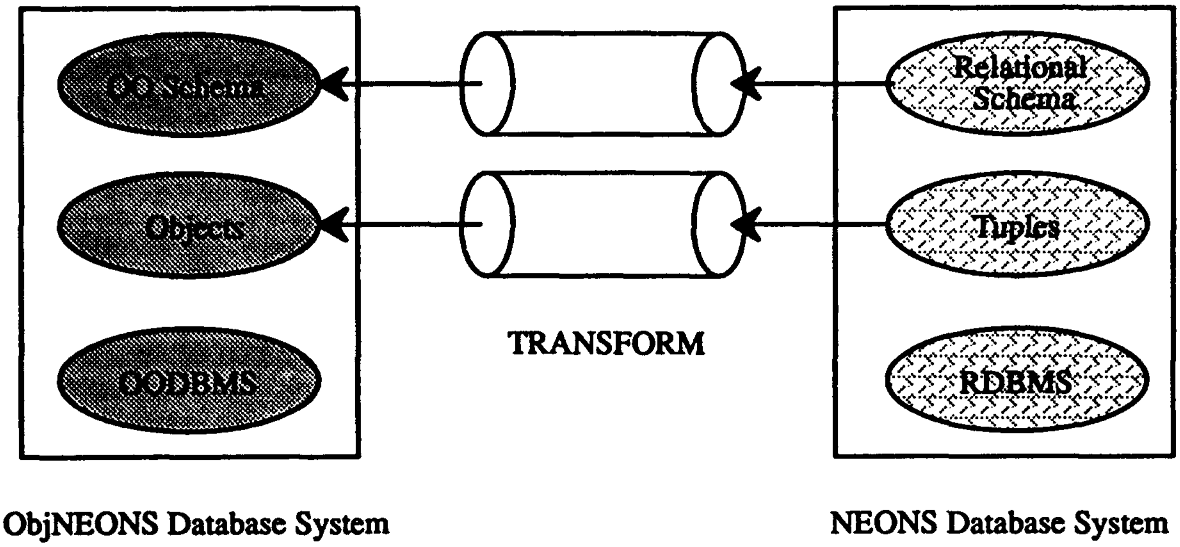

Figure 1.

The triangle represents the generalization/specialization relationship. A filled triangle indicates exclusive sub–classes while the white triangle represents inclusive sub–classes.

The primary goal of the object–oriented schema development was to exploit all the features provided by the object–oriented model to provide a view that represents the real world as much as possible. To that extent, the design of grid geometry and grid levels differ substantially (not surprisingly) from the corresponding relational schema. In the relational schema, the projection information is stored along with the registered geometry just as ordinary numbers and their interpretation (whether they represent 'minimum latitude' or 'row interval') is determined from the *descriptive realm* based on the projection type. We decided to make this explicit in the schema itself by enumerating all the projections as subclasses of a single *Projection* class so that no extra indirection is required to know more about the data (the metadata). It is stored in the schema itself. This ability to capture more data semantics is one of the biggest strengths of the object–oriented data model.

The transformation of relational tuples in NEONS to objects in ObjNEONS merits some elaboration. The transformation is not trivial in the sense that *not* every single tuple is a single object. The relational model stores the relationships implicitly in the form of foreign keys while the object model represents them explicitly in the form pointers or sets of pointers to the related object class. Therefore, whenever a tuple with a foreign key is being transformed into an object, a pointer to the related object (specified by the foreign key) must be fetched and assigned as part of the object's definition. ObjectStore supports the maintenance of relationships by automatically updating the links in one direction when the link is updated in the other direction. ObjectStore supports one–to–one, one–to–many, and many–to–many relationships.

Another important feature of ObjNEONS is that a lot of the metadata (stored in the *descriptive realm* in the case of NEONS) is stored along with the data itself. For example, it was observed that NEONS contains a lot data values that have units attached to them. While the *primary realm* contains a level value, say 13.5, the units of this value, say "milli–bars," is stored in the *descriptive realm*. It should be noted that there might not be a better way of doing this in the relational and conventional programming context without building in undesirable redundancy. To facilitate the unification of the data value and the units of measurement (the metadata), a new concrete object class called *Value* was defined. Each instance of the *Value* class contains a pair of values, namely, the numeric value and a character string representing the units. Also, the instances of *Value* may be used in any context where a numeric value is applicable. *Value* can also be made optionally 'intelligent' to prevent combining incompatible units in computations. *Value* instances are used extensively in ObjNEONS to represent latitude, longitude, levels, etc.

The approach taken for mapping the OO schema into implementation data structure in ObjectStore is as follows. In general, there are two components in a database: the *extension* and the *intension*. The *extension* of the database is the actual data (objects) in the database. The *intension* of a database "describes" the data without really enumerating the actual data. The schema of a database is a true representative of the *intension* of the database. In order to provide access to the *extension* of the ObjNEONS database, each class definition is augmented by an additional static data member called *extent*, which is a set (provided by ObjectStore) of pointers to all persistent instances of that class. Any class can be queried by examining the *extent* of each class. The many–end of a relationship is implemented by using sets of pointers. These sets of pointers are automatically maintained by the DBMS. This type of implementation is very convenient for providing fast responses to queries without any computational overhead (like *joins*) in the application program. For example, the object class GridGeometry contains a set of pointers to related GridDataset objects so that, given a particular geometry, all the corresponding data sets can be retrieved without any further query.

RESULTS: The schema for grid data was translated into ObjectStore C++ class definitions. Embedded SQL routines were written to read the data from the NEONS database and the data was used to build new objects that were consequently stored by using the OODBMS. New utility concrete class definitions were written and used (e.g., strings, date, time, epochal time, 'smart' values, etc.) for application development convenience. The implementation of the transformation of the NEONS data into ObjNEONS data is being carried out in phases. Currently, all the associative and descriptive data of NEONS has been stored in ObjNEONS. Work is in progress to store the data from the primary realm.

FUTURE RESEARCH: Future research should concentrate on developing a full–fledged implementation of ObjNEONS that should not only include other NEONS data types (viz., image, line, and llt) but also some newer types like the volumetric data type. Object–oriented technology offers immense potential. Efforts should be carried out to tap that potential to the maximum extent in several areas of software development. Sridhar Koduri is working on another project to develop a user interface for the grid data stored in ObjNEONS object–oriented database. The analysis and design of the interface will involve consultation with the scientists at CAST to ensure that the resulting interface will provide an easy–to–use environment for grid data.

Some new work could be done to study the suitability and applicability of ObjectStore's version handling capability. Version management allows multiple versions of the same object to be stored in the database. This will help the data to evolve naturally while hiding the intermediate object states from other transactions.

RESEARCH ADVISOR: Dr. Julia Hodges, MSU Department of Computer Science

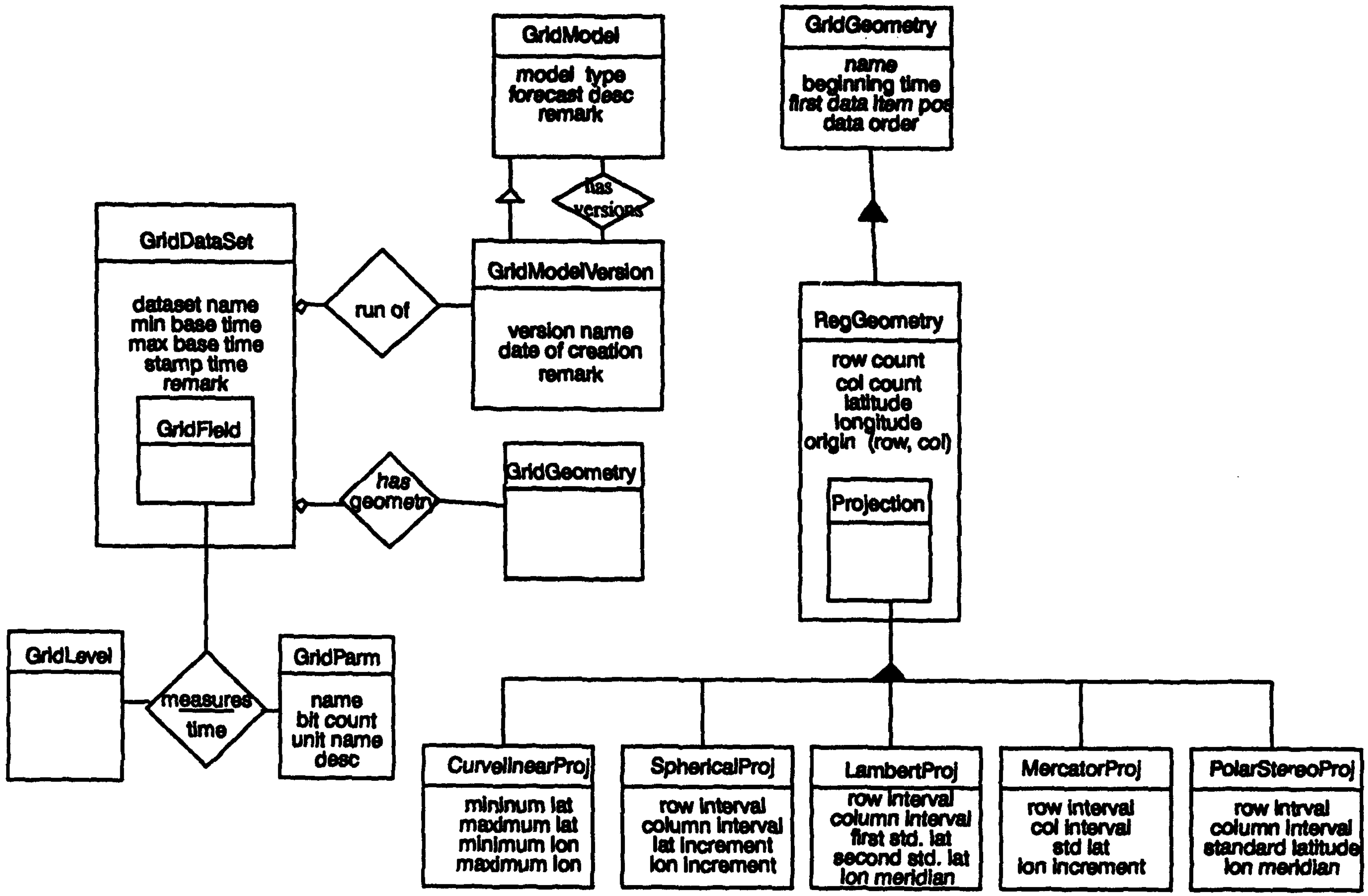

Figure 2. Object Schema for Grid Data

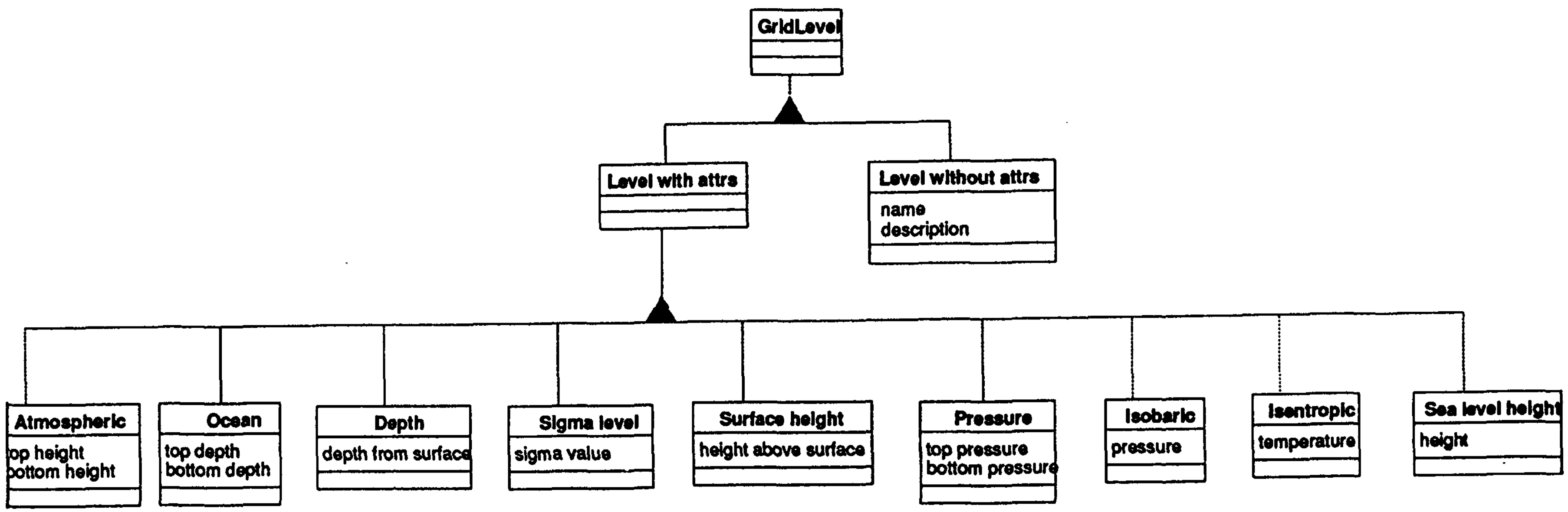

Figure 2. Continued

An Object–Oriented Prototype for a Geophysical Database
Subtask: An Object–Oriented Database Schema for Image Data

Koduri Sridhar
Julia Hodges
P.O. Drawer CS
Mississippi State, MS 39762
{sri,hodges}@cs.msstate.edu

Project Title: Object–Oriented Database Schema for Image data

Objective: The objective is to develop an object–oriented schema for the image data type in NEONS.

Approach: The database schema for the object classes and relationships needed to represent image data was developed using the Fusion object–oriented software development methodology. Fusion divides the development process into analysis, design, and implementation[Coleman et al. 1994]. In the analysis phase, the developer defines the intended behavior of the system. The concepts that exist in the domain of the problem and the relationships between them are captured in the object model in this phase. In the design phase, the developer chooses how the system operations are to be implemented by the run–time behavior of interacting objects. During this phase operations are attached to classes; the developer also chooses how objects refer to each other and what the appropriate inheritance relationships are between classes. The substructures of all the classes and their operations are investigated in detail[Coleman et al. 1994]. During the implementation phase, the developer turns the design into code in a particular programming language and DBMS; in our case; this is C++ and ObjectStore. The notation of Fusion allows the systematic discovery and preservation of the object structure of the system.

The analysis and design of the object–oriented schema for image data involved consultation with the scientists at CAST to ensure that the resulting database will provide a more natural representation of the data. The analysis phase and part of the design phase of the project are complete at this point. In the analysis phase classes of objects in the database system were identified. Special care was taken in identifying the conceptual objects as well as the real objects. The real world associations that exist or conceptual relationships between these different classes were identified. The stronger associations were converted to aggregations. Based on the semantics of the data, the participation and cardinality of the classes in the associations were determined. A cardinality constraint restricts the number of objects which may be associated with each other in a relationship. All objects in the adjacent class must appear in the relationship; this is indicated by a total marker.

We had a meeting with our CAST contact person, Mr. Valentine, to review the design and the semantics of the data in the database schema. He provided input on the details of the image data and the refined system requirements. Based on our discussions in the meeting and the database design document for NEONS, we developed the object model for the image data in a refining process using Fusion notation and methodology.

The object model notation of Fusion is based on an extended entity relationship notation. It can represent classes, attributes, relationships, aggregation, generalization, cardinality, and participation[Coleman et al. 1994]. The boxes are the classes and the diamonds are the relationships. Each class is represented by a box, with the name of the class at the top, separated from the rest of the box by a line. The names of the attributes belonging to the class are named below the line and are enclosed in the box of each class. The relationships are shown as a diamond joined to the participating classes by arcs. The aggregation (the class which has component classes) is represented by nested boxes. The notation for a generalization (is–a or a–kind–of relationship) is a filled triangle connecting a supertype to subtypes. The cardinalities (1 – one, + – one or more, * – zero or more) are represented by annotating the arcs connecting the relationship to the classes. Small square filled boxes represent total participation of the class in the relationship.

Results: The object model which defines the static structure of the information in the image database schema (shown in figure 1) was designed and developed.

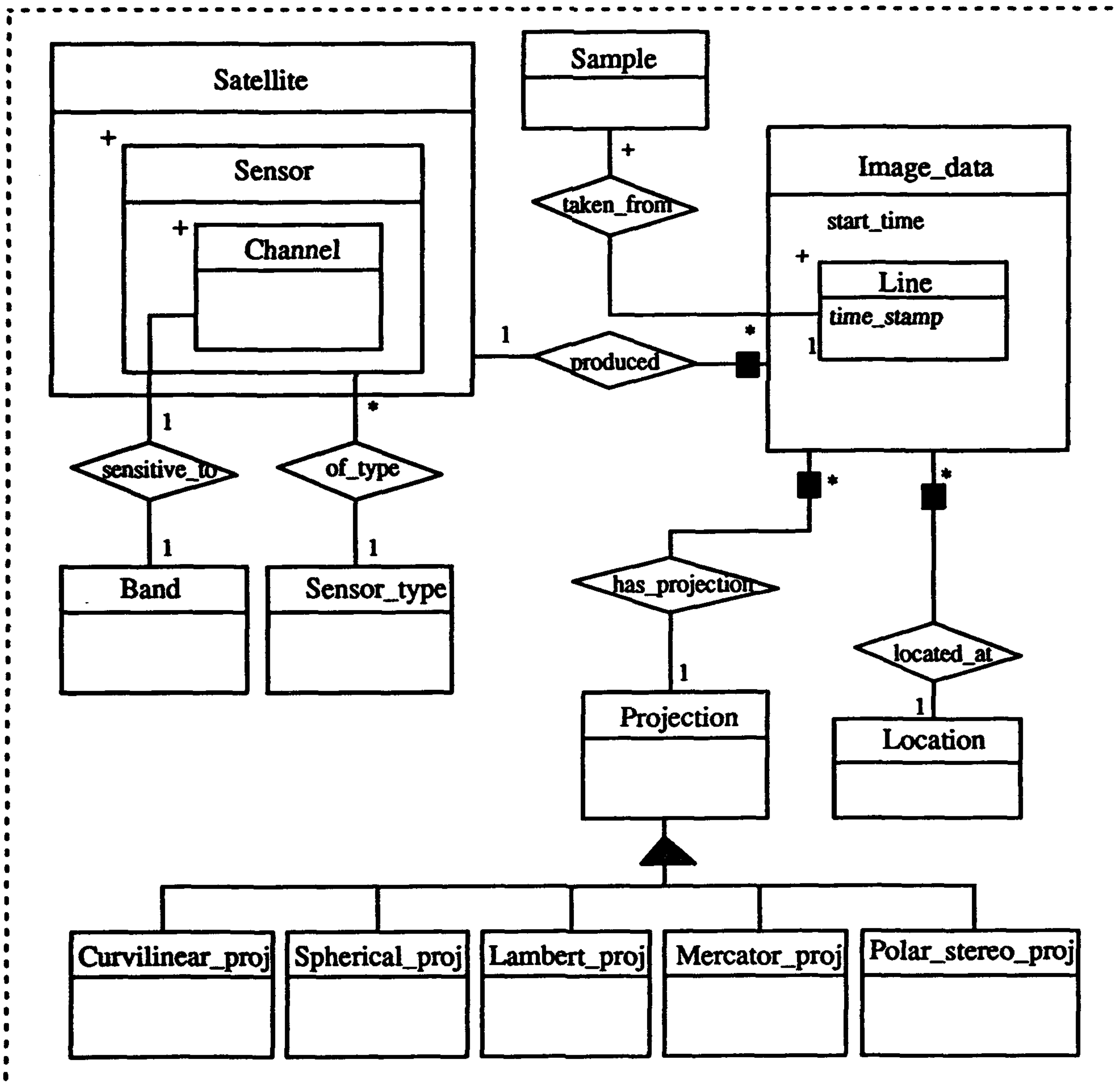

Figure 1. Object Model for Image data

Future Research: We plan to implement prototype object–oriented database for the image data using ObjectStore – a premier object–oriented database management system.

Reference: Coleman, Derek., et al. 1994. *Object–Oriented Development – The Fusion Method.* Prentice Hall, Englewood Cliffs, New Jersey 07632.

Research Advisor: Dr. Julia Hodges, MSU Department of Computer Science

******

An Object–Oriented Prototype for a Geophysical Database
Subtask: Evaluation of Commercial Object–Oriented DBMSs

Chandrashekar Ramanathan
Koduri Sridhar
Julia Hodges
P.O. Drawer CS
Mississippi State, MS 39762
{shekar,sri,hodges}@cs.msstate.edu

PROJECT TITLE: Evaluation of Commercial Object–Oriented Database Management Systems (ODBMS)

OBJECTIVE: To evaluate commercially available OODBMSs and recommend for purchase systems that appear to best fit CAST's database needs.

APPROACH: Selecting an ODBMS for purchase is a complex task. We gave careful consideration to several aspects before arriving at a decision. For our study, we considered nine popular ODBMS products, namely, GEMSTONE, M.A.T.I.S.S.E, MONTAGE, OBJECTSTORE, ONTOS, OBJECTIVITY/DB, POET, UniSQL, and VERSANT. A major factor in choosing these nine products for evaluation was that all of them were C++ compatible, which was rated as the most important requirement by CAST. We also ensured that the products that we considered would run on the platform specified by CAST (a SUN Sparc running under Solaris).

The main features of interest in any ODBMS are the primitive data types and the built–in aggregates; support for composite/complex objects, dynamic schema evolution, version management, long transaction management, lock management, and authorization; and the interface tools. We prepared a list of desirable and required features that the ODBMS for CAST should support and asked the vendors to tell us whether their system supported them or not. This was done by preparing a questionnaire and sending it to the vendors. The questionnaire is attached at the end of this report. The responses from the vendors, the information from computer science literature, literature provided by the vendors, and some published reviews of the products were combined to get a consolidated picture of all the systems under review. These results, compiled in the form of an information grid, are attached at the end of this report. Following is a brief description of the various systems under consideration.

Different products support different machines and operating systems. GEMSTONE is a product of Servio Corporation. The system is well–suited for use in multi–user, multi–platform client/server applications. M.A.T.I.S.S.E. is a heterogeneous cross platform client/server architecture–based ODBMS developed by ADB, Inc. MONTAGE is the commercial version of the well–known Postgres DBMS (first developed at the University of California at Berkeley) and is produced by Montage Software, Inc. MONTAGE is a fully relational database management system having extensions to support object–oriented concepts and models. OBJECTSTORE, from Object Design, Inc., is an ODBMS that provides a tightly integrated object–oriented programming interface using C++. It has a client/server architecture where each workstation can simultaneously access multiple databases at many servers. A product of Objectivity, Inc., OBJECTIVITY/DB has a fully distributed client/server architecture that transparently manages objects across heterogeneous environments and multiple databases. ONTOS, a multi–client, multi–server distributed database management system, is a product of Ontologic, Inc. POET (Persistent Objects and Extended database Technology) is a product of BKS Software that has a tight semantic integration with C++. UniSQL is a unified relational and object–oriented database system produced by UniSQL, Inc. Finally, VERSANT is a client/server OODBMS that is compatible for distributed, multi–user applications.

We had requested our CAST contact person, Mr. Ramesh Krishnamagaru, to send us some input on the system requirements and preferences on a scale of 0 (not important) to 4 (most important) on various features of ODBMS. Our evaluation of the nine products was based on the importance assigned to the features vis–a–vis the products support for the several important features (e.g., price, language support, interface, technical aspects, etc.).

RESULTS: The systems we evaluated can be broadly classified into two categories: those that are built by extending C++ to handle persistent objects and those that are built by extending the relational model to handle hierarchies of objects. OBJECTSTORE, VERSANT, POET, OBJECTIVITY/DB, ONTOS, and M.A.T.I.S.S.E. come under the former category. UniSQL and MONTAGE come under the category of extended relational databases. GEMSTONE is neither an extended C++ nor an extended relational system. It is recognized as the most nearly pure object–oriented DBMS and is one of the oldest commercial ODBMS products. Based on our evaluation, we felt that CAST's needs can best be met by one of these three products (in decreasing order of preference): OBJECTSTORE, UniSQL, and GEMSTONE. Following is a brief critique of all the systems based on the criteria we had set that led to our final recommendations.

Among the extended relational database systems, we found that MONTAGE does not allow methods for the classes to be written in C++ (although it allows C++ applications to access the database), a factor that eliminated it from consideration. On the other hand, methods in UniSQL can be written in any compiled language. Looking at the information grid, it can be seen that UniSQL meets all our evaluation requirements fairly well. Also, it comes with an extended SQL language, which might help all the current users of relational databases at CAST to get started quickly on using the system. However, it should be noted that the ease and naturalness with which C++ applications could be developed will be known only after actually working with the system. This is a point of concern with UniSQL because it was not developed by extending an object–oriented programming language. Also, it is not clear how well UniSQL maintains referential integrity because, for example, it cannot handle exclusive components of composite objects.

We recommended GEMSTONE because GEMSTONE is developed based on the Smalltalk language's programming model, which is considered to be one of the 'pure' object–oriented languages. Therefore, we think that the full benefits of object–orientation can be realized by using such a system. Note that the system itself is language independent in the sense that it doesn't matter whether objects are created by applications written in Smalltalk or C++. Once again, it is not clear how easy it is to make C++ applications communicate with Smalltalk environment. One individual who is currently involved in trying to do this has told us that it appears to be awkward, requiring a lot of preliminary bookkeeping.

We think that the ability to maintain multiple versions of conceptually the same object (version management) should prove to be useful in the long term for CAST applications. In fact, we intend to explore how versions can be useful in the prototype for geophysical databases we are building. Therefore, based on the ability of the systems to support versions, we have excluded ONTOS and POET, both of which support neither object versions nor schema versions. Among the products, POET and VERSANT provide poor database authorization control. Also, VERSANT relies on the operating system for disk media protection. From the available information, POET does not have any non–programmatic interface. Despite our repeated requests for information from the vendor about OBJECTIVITY/DB, we did not have enough information for comparison at the time of doing the evaluation. M.A.T.I.S.S.E. has some limitations comparatively in the areas of transaction processing and also in the provision of built–in aggregates. We decided to include OBJECTSTORE as one of the choices because of its neat and tight integration with C++ and our observation that OBJECTSTORE is one of the most popular and widely known products in the area. One of its strengths is to treat type independent of persistence. That is, any conceivable C++ object can be made persistent with little change to the corresponding transient object's definition. It also meets most our evaluation criteria fairly well. The OBJECTSTORE vendor provided us with the name of a user who had implemented an image database using OBJECTSTORE. The information that we got form this user indicated a high degree of satisfaction with the product. In general, OBJECTSTORE should meet all the requirements very well.

After we completed our evaluation and submitted our recommendation to CAST, they purchased a copy of OBJECTSTORE. It has been installed in one of the research laboratories in the Department of Computer Science at Mississippi State University.

FUTURE RESEARCH: An object–oriented database for the grid data by using the OBJECTSTORE system is already under development by Chandrashekar Ramanathan. We also plan to store in the object–oriented database other data types such as the image data, the llt data, and the line data currently in the NEONS relational database.

RESEARCH ADVISOR: Dr. Julia Hodges, MSU Department of Computer Science

Questionnaire on ODBMS for Vendors

I. Which of the following platforms does your oodbms support?

Sun SPARC running UNIX/SOLARIS 2
Sun SPARC running SunOS 4.1

II. Please provide the following information concerning costs.

1. What is the cost of purchasing your system?
2. Do you offer an education discount? If so, please briefly describe the terms and the amount of the discount.
3. Please briefly describe the cost and terms of the technical support that you provide for your oodbms.
4. What provision do you have for upgrades?

III. Please mark with an X those features that are supported by your system.

1. Primitive data types
   - real
   - integer
   - string
   - character
   - binary objects
   - images
   - others (please specify)
2. Aggregates
   - sets
   - bags
   - lists
   - arrays
3. Complex objects
   - exclusive components
   - shared components
4. Automatic referential integrity (relationships supported)
   - 1:1
   - 1:many
   - many:many
5. Versions
   - support for object versions

support for automatically merging object versions
schema versions

6. Interactive tools available for:
   creation/change of schema
   instantiating objects
   DBA operations

7. Schema evolution
   add a new attribute to a class
   drop an attribute
   change name of a method
   change code of a method

8. Transaction properties
   long transaction (hours, weeks, months, etc.)
   shared transactions

9. Granularity of locks
   data page
   index page
   a single class within a page
   a single instance within a page
   composite object

10. Authorization control
    database level
    class level
    attribute or data member level

11. Disk media protection
    dual copy disk mirroring
    deferred copy

12. Backup/Recovery
    tools to backup database

13. Distributed databases

14. Did anybody develop a geophysical application with the system?

15. Any other additional information

16. C++ compiler compatibility
    cfront 2.1
    cfront 3.0
    gcc 2.4.5

ODBMS Information Grid

	Gem-stone	Montage	M.A.T.I. S.S.E.	OBJEC-TIVITY/ DB	OB-JECT-STORE	ONTOS	POET	UniSQL	VER-SANT
☐ **Platforms supported**									
Sun Sparc running UNIX/SOLARIS2	Y	Y	Y	Y	Y	Y	Y	Y	Y
Sun Sparc running SunOS 4.1	Y	Y	Y	Y	Y	Y	Y	Y	Y
☐ **Educational discount availability**	Y	Y	80%	Y	Y	100%	50%	60%	80%
☐ **Technical features**									
• **Primitive data types**								Y	
Real	Y	Y	Y		Y	Y	Y	Y	Y
Integer	Y	Y	Y		Y	Y	Y	Y	Y
String	Y	Y	Y			Y	Y	Y	Y
Character	Y	Y	Y		Y	Y	Y	Y	Y
Binary object	Y	Y	Y				Y	Y	Y
Image	Y	Y	Y			Y	Y	Y	Y
Others	image sound	'blades' like doc, spatial image				audio	date time	audio	date time
• **Aggregates**									Y
Sets	Y	Y	N		Y	Y	Y	Y	Y
Bags	Y	N	N		Y	N	N	Y	Y
Lists	Y	N	Y		Y	Y	N	Y	Y
Arrays	Y	Y	Y		Y	Y	Y	Y	Y
• **Complex objects** (Preference: 3)	Rating: 1			Y	Rating: 4	Rating: 2			Rating: 2
Exclusive components	Y		Y		Y		Y	N	
Shared components	Y		Y		Y		Y	Y	

	Gemstone	Montage	M.A.T.I. S.S.E.	OBJECTIVITY/DB	OBJECTSTORE	ONTOS	POET	UniSQL	VERSANT
• **Relationships supported explicitly**									
1:1	Y	Y	Y	Y	Y	Y	Y	Y	Y
1:many	Y	Y	Y	Y	Y	Y	Y	Y	Y
many:many	Y	Y	Y	Y	Y	Y	Y	Y	Y
• **Distributed database**	Y	N	N	Y	Y		N	Y	Y
• **Versions** (Preference: 2)	Rating: 0	Y			Rating: 9	Rating: 0			Rating: 9
Support for object versions	Y	Y	Y	Y	Y	N	N	Y	Y
Support for automatically merging object versions	Y	N	N		Y	N	N	Y	N
Support for schema versions	Y	N	Y	Y		N	N	Y	N
• **Interface tools available for**	Rating: 10 ST* 2 OPAL				Rating: 6	Rating: 5			Rating: 5
Changing schema	Y	Y	Y				Y	Y	Y
Instantiating objects	Y	Y	Y					Y	Y
DBA operations	Y	Y	Y	Y			Y	Y	Y
Others	GeoDE (Visual Prog.)			report generator		STUDIO DB Designer		GUI–builder	Versant View
• **Graphical User Interface**	Y	Y	Y	Y	Y	Y		Y	Y
• **SQL support**	N	Y	Y	Y	Y	Y	N	Y	Planned
• **Schema evolution** (Preference: 2)	Rating: 5			Y	Rating: 2	Rating: 4			Rating: 3
Add a new attribute to an existing class	Y	Y	Y		Y		Y	Y	Y
Drop an existing attribute	Y	N	Y		Y		Y	Y	Y
Change name of an attribute	Y	Y	Y				N	Y	Y
Change domain of an attribute	Y	N	Y		Y			Y	Y
Add a new method to a class	Y	Y	Y		Y		Y	Y	Y

	GEM-STONE	MON-TAGE	M.A.T.I. S.S.E.	OBJEC-TIVITY/DB	OB-JECT-STORE	ONTOS	POET	UniSQL	VER-SANT
Change the name of a method	Y	N	Y		Y		Y	Y	Y
Drop an existing method from a class	Y	Y	Y		Y		Y	Y	Y
Change the code of a method	Y	Y	Y		Y		Y	Y	Y
• **Transaction Properties**									
Long transactions	N	N	N	Y	Y				Y
Shared transactions	Y	Y	N	Y			Y		Y
• **Granularity of locks**									
Data page		Y	N		Y			N	N
Index page		Y	N					N	N
A single class within a page		Y	Y					N	N
A single instance within a page		N	N		Y			Y	Y
single object	Y	?	N	Y			Y	N	Y
composite object	Y	N	N	Y			Y	N	Y
• **Authorization control**									
database level	Y	Y	Y		Y		N	Y	Y
class level	Y	Y	Y				N	Y	N
attribute or member level	Y	Y	Y				N	Y	N
other							O.S.		via meth-ods
• **Disk media protection**									
dual copy disk mirroring	Y		N		Y		N		N
deferred copy	Y		N				N		N
other		O.S.	Replica-tion at the disk level.		Logs		O.S.	active and archived logs	O.S.
• **Backup/Recovery**									
tools to backup database	Y	Y	Y	Y	Y		N	Y	Y

	GEM-STONE	MON-TAGE	M.A.T.I. S.S.E.	OBJEC-TIVITY/DB	OB-JECT-STORE	ONTOS	POET	UniSQL	VER-SANT
❑ Language compatibility									
cfront 3.0		N	Y	Y	Y		N	Y	Y
cfront 2.1		N	Y		Y		Y	Y	Y
gcc 2.4.5	N	N	Y		N		N	N	N
❑ Geophysical applications		Y			Y		N	Y	Y
❑ Additional information		Commercial Postgres; all user defined functions in SQL or C.	Supports triggers at object, attribute, and relationship level.						

* The given ratings for Objectstore, Ontos, and Versant were obtained from Ahmed et al. (1992).

Symbol legend

Y – Yes
N – No
blank – not known

Hollie Molyneux
M.S. Program, Department of Mathematics
University of Southern Mississippi

Title: Test and Evaluation of the CAST Model Evaluation System (CMES)

Objective: The objectives were to gain an understanding of the operation and capabilities of the CMES, and using designated oceanographic data sets, evaluate the utility and friendliness of the graphical user interface, and the utility of the CMES as a tool for ocean model development.

Approach: The initial effort was in learning how to select data and use the visualization mode (VISMOD) by experimenting with the various data sets already residing in the CMES database. Following this, bathymetry and water quality data sets from the Back Bay of Biloxi were read into files and formatted for ingestion into the CMES. It was necessary to remove extraneous symbols, and to develop programs to write the data into columns and convert minutes and seconds of latitude and longitude into degrees. Once formatted, the data sets were ingested. The CMES was then used to interpolate the data to a grid and export//display the results. The other approach was to run DieCAST model to simulate a 30-day period in the Gulf of Mexico at 20 levels. Since the grid was already in the databases, model output could be imported directly into the CMES and did not have to be ingested. A program developed by Dr. Harsh Anand of CAST was modified to accommodate a 30-day period with 20 levels. The resulting file was imported into the CMES and the data displayed.

Results: VISMOD was found to be very self-explanatory and user-friendly. The broad range of options allowed the data to be displayed in a form well suited to the parameter of interest. For example, when the change in a parameter was very gradual, it was somewhat difficult to see the color distinctions using the topographic hues. However, switching to rainbow tones made the distinction clear. In the case of 2-D graphics, changing the date or level to be viewed was a simple procedure done directly within VISMOD. The ability to overlay grids and coastlines were also useful. Interpolations and the import tool were straightforward, but could benefit from the inclusion of help screens or a user's manual. For example, depths must be specified for interpolations. It is possible to choose standard depths, user-specified depths, or user depths from a file. However, it was not clear what the standard depths were, or what format would be required. When interpolation was completed, saving the results for display as gridded data and exporting it to a file required only clicking on the save or export buttons and specifying a filename, making these features very convenient. Overall, the CMES proved to be an excellent tool for visualization of the designated oceanographic data sets. With simple format changes, it easily accepted interpolated and displayed the data. The DieCAST model also ran successfully on the CMES. Formatting the output for importation required modification of an existing program. The model's output was successfully imported into the CMES and the results displayed. Figures 1 and 2 show pressure and temperature, respectively, on day 30 at the top level. It would be advantageous to be able to setup, execute, analyze DieCAST runs directly from the CMES, without the necessity of intermediate programming steps. The modified program could serve as the basis of such a CMES addition.

Future Research: For the CMES to be useful and accessible, help screens and a user's manual describing all options should be prepared. To make the CMES more convenient and self-contained, modifications should be made which would allow models to be setup, executed, and analyzed directly.

Research Advisor: Dr. Louise Perkins, Department of Scientific Computing, University of Southern Mississippi.

Figure 1. DieCAST Model Output of Temperature on Day 30.

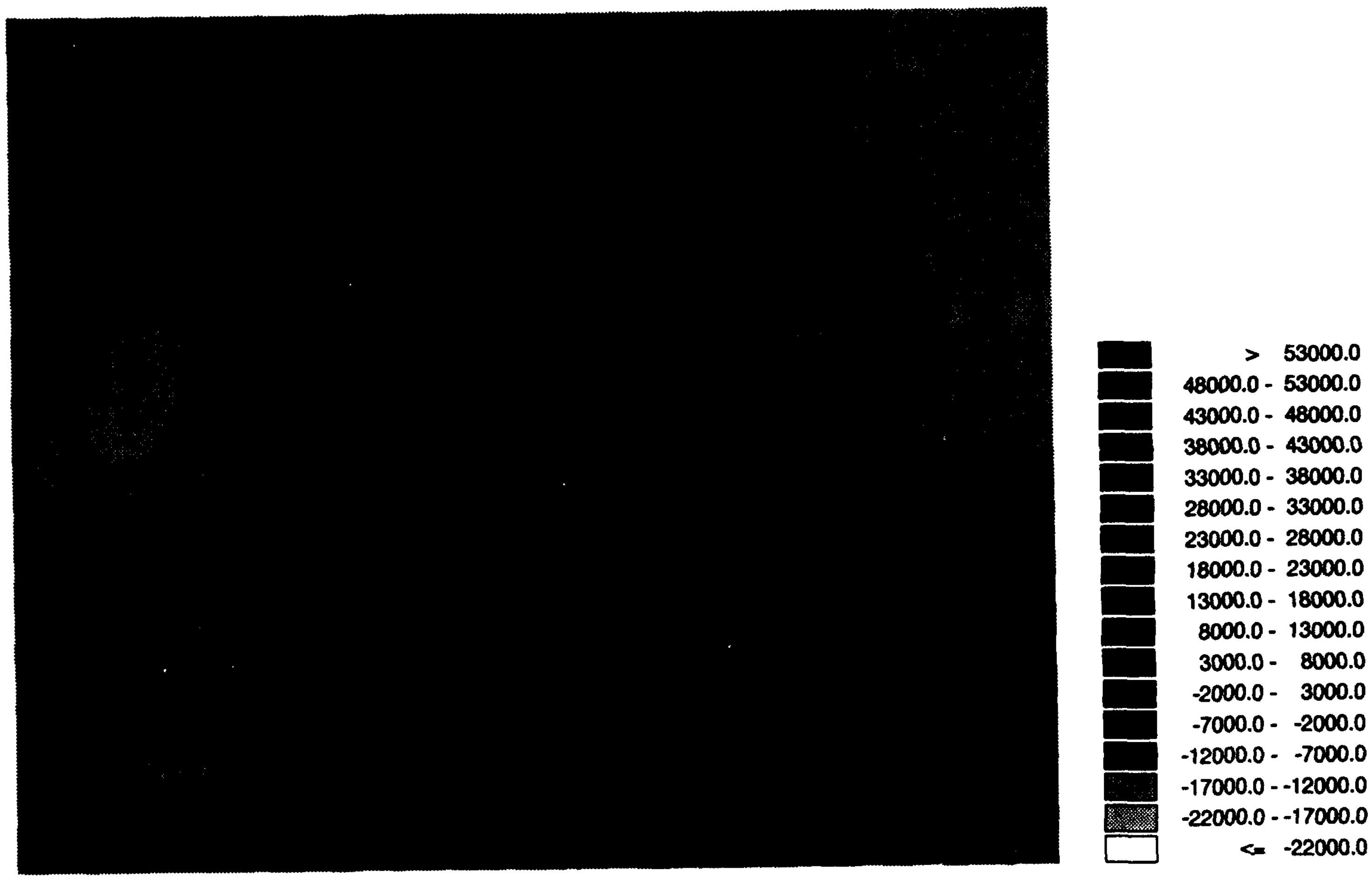

Figure 2. DieCAST Model Output of Pressure (10^5 dynes/cm^2) on Day 30.

Mickey L. Barton
B.S. Program, College of Science and Technology
University of Southern Mississippi

Project #1 Title: Enhancement of the Naval Interactive Data Analysis System (NIDAS) In Viewing Multiple Fronts and Eddies

Objective: The analysis of fronts and eddies includes a comparison of their changes over time. It is easier to make these comparisons when the fronts or eddies are viewed simultaneously. The objective of this project was to give the user the ability to view multiple front or eddy data in one plot using NIDAS. The user was to be given the ability to select one, multiple, or all data from the pick list, as well as to display this data in separate colors when needed.

Approach: The interaction of the NIDAS functions related to the selection and display of data was first studied. The approach was to see how the selection processes were implemented, then see how to implement the selection of multiple fronts/eddies and colors.

Results: The OSF/MOTIF widgets associated with the selection of fronts and eddies were modified, as were widgets associated with the selection of color. NIDAS now gives the user the ability to display one, multiple, or all fronts or eddies. The user also has the ability to display fronts and eddies in multiple colors when he/she selects up to seven fronts or eddies. If the user selects more than seven fronts and eddies, they are displayed in one color only.

Research Advisor: Mr. Dharmesh Krishnamagaru, Center for Air Sea Technology, Mississippi State University.

****

Project #2 Title: Investigation of a Window Environment for Retrieving and/or Viewing Oceanographic Data Over the Internet

Objective: Large amounts of data are being collected over the worlds bodies of waters. Scientists need to be able to access this data for use with their models and other projects. One way is for them to access the data via the Internet. They also need the capability to view the data before retrieving it, so as not to waste time downloading that which is not suitable for their project.

Approach: The GMT plotting system (a freeware product) was first investigated to determine its applicability for use as a plotting tool. A vehicle was also needed to allow access to retrieve and/or view the data in the database. Mosaic was studied for its use as this Internet vehicle.

Results: The GMT plotting system was found to be adequate as the system plotting tool. We determined that Mosaic was an excellent way to allow the user access to the data and to plot it, if so desired. Once a user is connected he may connect to our Mosaic server. Upon selecting the plot ordering form the user has access to ordering the data. Once the user has entered the attributes requested, he may then view and/or retrieve data.

Research Advisor: Mr. Valentine Anantharaj, Center for Air Sea Technology, Mississippi State University.

Owen Lagarde
M.C.S Program, Department of Computer Science
Mississippi State University

Title: Exportation and Extraction Capability for Remote Users of a Navy Environmental Operational Nowcasting System (NEONS) Database

Objective: Initial objectives were defined by requirements of the Tactical Oceanography Wide Area Network (TOWAN); volumetric data sets stored on the TOWAN system must be accessible and retrievable to a remote user's file system via a connecting network (presumed Internet). This capability could be achieved through two independent methods -- a command line interface and a graphical interface (as an extension of the CAST BROWSER, an existing NEONS database access tool).

Secondary requirements included the following:

- The command line interface was to use standard input and output streams, allowing semiautonomous batch processing.
- The graphical interface was to be as intuitive, data independent and code-modular as possible, allowing easy incorporation of data types other than volumetric.
- The selection scheme was to allow retrieval of contiguous subsets of a model field ranging in size from one data sample point to all points in the specified field.
- Output was to be written to the local file system, either as space- and linefeed-delimited ASCII text, or as NCSA Hierarchical Data Format objects (NCSA HDF 3.3r3) in which the field data would be considered as a three dimensional block of values and stored as a series of profiles; three profiles would be used to store spatial attributes of the field from which parameter data was extracted, and one additional profile (for each extracted field parameter) would be used to store parameter data values.

Approach: The project commenced with a study of X-Motif programming, EMPRESS imbedded SQL operations, CAST BROWSER design and mechanics, and NEONS design and support software mechanics. A command-line version was developed in FORTRAN and C using NEONS function calls. Routines for non-interactive retrieval of array data from within the BROWSER package were designed and implemented such that a user interface allowing selections of array data subsets could be added with a single source code call. The BROWSER package, with its new ability to retrieve full arrays from an EMPRESS-managed NEONS database, was then used to verify the retrieval output against console output from an EMPRESS command line interface. Verification was followed by tentative validation of output formats and design of a user interface for selection of array data subsets. The user interface was gradually modified to include the following user options and abilities:

- Point Selection Area -- users would be presented with a graphic window in which selections could be made with a graphic pointer device (presumed a mouse or lightpen); the array would be considered as evenly filling the window and the user would be able to "click-and-drag" the pointer within the window to select horizontal subsets of the array.
- Level Selection Area -- users would be presented with a list of level numbers and values and be able to "click-and-drag" the pointer within the list to select vertical subsets of the array.
- Latitude/longitude Grid Display -- users would be able to enable or disable display of a horizontal grid that would denote the horizontal boundaries of the selected array and label intermittent points of latitude and longitude along the array's major axes.
- Point Display -- users would be able to enable or disable display of a highlighted pixel at the relative latitude-longitude location of array data points in the selection window.
- Coastline Display -- users would be able to enable or disable display of coastlines in the selection window that lay within the area of the selected array.
- Point Selection Modes -- users would be able to choose from a series of modes which, in turn, would alter the manner in which the user's selection action was interpreted. Three modes were requested by the clients: "Area" -- the pointer (via a "click-and-drag") would define a rectangle over the array's horizontal surface and select all points within that rectangle. "Point" -- the pointer

would define a location in the array and the single array data point closest to that location would be selected. "Great Circle Track" (also known as "Great Circle Path") -- the pointer would define two locations over the array's horizontal surface to be used as the control points of a curve (Great Circle Path (GCP) definition, as per Navy standards). Array data points closest to the curve of the GCP would be selected (per Naval Research Laboratory GCP volumetric array data selection standard).

- Keyboard Entry of Array Boundaries -- via text input cells, the user would be able to enter the latitude and longitude of the current pointer location (and pointer starting position, if required) via the keyboard, as an alternative to manually positioning the pointer and dragging to define the area. Typed numerical input would be accurate to six decimal places and, since the values would be interpreted as point locations, they would be independent of field orientation, the coordinate system, range, location, or entry order. Upon receiving input location values, the interface would update the display as defined by the currently enabled display options.

Results: The prototype was designed and installed in the BROWSER and activated by a menu button in the volumetric support area. A user, having selected a specific volumetric array with the current BROWSER capabilities, can then select a "Retrieve" button from the "Options" menu and launch a selection tool as described above. To retrieve array data from the database, the user 1) chooses a point selection mode from the "Mode" menu bar item (default "Area"), 2) specifies the boundaries of the subsection of the array (either by "click-and-drag" in the Map Display Window, or by entering the latitude and longitude of the boundary coordinates into keyboard entry textboxes), and 3) specifies the level (vertical) boundaries of the array in the Level Selection Area (by "click-and-drag" within the list of levels). If users wish a more exact specification of the boundaries of usable data or the locations of specific data points, they may select the "Options" menu bar item and toggle the grid, coastlines, and point display controls as desired.

After verifying output of the data retrieval/extraction code, object code modules of the BROWSER were re-linked with the CAST NetNeons library to allow the BROWSER to access NEONS databases via a CAST-developed network server which supports remote access from client sites.

In general, the BROWSER system may be configured as follows:

- A "provider" of a NEONS database can launch an instance (child process) of the NetNeons server using a port on their host machine. The server inherits the access privileges of the user's account. The "provider" may also post (via electronic mail or other means) the location of the server (Internet address of the host machine and port number used by the server) and general information about data available via the server.
- Prospective clients can access the server port and launch an instance of the BROWSER (compiled with the NetNEONS client library).
- Interaction between the client's instance of the BROWSER and the provider's NEONS database appears as if both processes were running on one hardware system; inter process communication is transparent to the user (except for network bandwidth/loading considerations).

Future Research: The prototype was developed for the NEONS volumetric data type to satisfy the immediate needs of TOWAN project participants. The interface, however, is largely independent of data organization, type, and range, and assumes only that latitudes and longitudes increase from west to east and south to north, respectively. It should therefore be relatively straightforward to modify the system to support other NEONS data types (grid, latitude/longitude/time, line and image). The grid data type is a particularly good candidate since it was the precursor of the volumetric data type. Furthermore, the interface's only requirement for the BROWSER is for identification of a unique volumetric data array. This requirement could be equally satisfied by an extension of the extraction interface, making the retrieval code an independent application.

Research Advisor: Mr. Michael S. (Steve) Foster, Center for Air Sea Technology, Mississippi State University.

Michael S. Baer, IV
B.S. Program, USM Cooperative Education Program
University of Southern Mississippi

Project #1 Title: Implement a Geophysical Database Using the Oracle Relational Database Management System (RDBMS)

Objectives: To reconfigure existing geophysical applications, built using the Empress monolithic RDBMS, to use the Oracle client server RDBMS.

Approach: Install the Oracle database and configure it to work with existing hardware. Then acquire skills to create, maintain, and administer the RDBMS. The next step is to create the database so that data will be stored logically and efficiently. After these tasks have been accomplished, NEONS software developed to run on the Oracle RDBMS must be installed on the database. Next, CAST upgrades used on NEONS for Empress, must be added to this new NEONS software package. Lastly, existing database applications must be ported to use this new software.

Results: Oracle was installed at CAST on a Sun Sparc 1 Workstation. Extensive knowledge of the Oracle client server RDBMS has been gained at CAST by hands on experience and experimentation. NEONS for Oracle has also been installed on this system and, after altering Oracle NEONS to incorporate all of the CAST upgrades, a CAST Data Browser was ported to use the Oracle NEONS software. The Browser port was successful and other CAST applications will now be ported to use this software.

****

Project #2 Title: Flat File Feature of Real Time Wave Forecasting System (RTWF)

Objective: To create a function on the RTWF that will allow the user to import data from a flat file instead of the database.

Approach: Develop an algorithm that will efficiently search flat files. To increase efficiency, the algorithm will eliminate unnecessary file reads. This algorithm will then be incorporated into a C language function. After testing and evaluation, this function will be included in the RTWF application.

Results: An algorithm was developed that allowed the flat file to be searched with a minimum number of file reads. After testing, the application was incorporated into the RTWF application.

****

Project #3 Title: The World Weather Watch (WWW) Chartwall

Objective: To simulate a chart wall of maps on a computer by creating a graphical user interface (GUI) that contains icons of these maps.

Approach: To develop this, the X-Window system using the OSF Motif toolkit was studied to gain the knowledge necessary to incorporate these tools into the application. After a thorough understanding of X has been achieved, the developer then creates a program to convert the weather maps from their TIFF format to an Xpixmap format that may be placed on top of push button widgets. Pushbuttons widgets with these images on them will then be created in a GUI in such a way that when the buttons are selected by a user, they will call up a routine to display the original weather map in its full screen version.

Results: The application was developed, tested, and has been incorporated into the WWW application.

****

Project #4 Title: CAST Model Evaluation System (CMES) Documentation

Objective: To develop documentation for the CMES using Mosaic software.

Approach: To obtain an understanding of CMES and its various applications, and then to create an online (internet) user manual for CMES using hypertext linked documents in Mosaic. Documents which explain all aspects of CMES will be created, and divided into logical sections. These sections each represent one Mosaic page. On these pages there will be graphics and highlighted words. The highlighted words will be hyperlinks to other pages. X window dumps of various CMES user screens will be created and the graphics will be inserted into the hyperlinked pages. These graphics will then be given "hotspots" that will act as hyperlinks to other pages. These features will make the documentation interactive.

Results: A full understanding of CMES has been acquired and clear concise documentation for CMES has been developed using Mosaic.

Research Advisor: Mr. Ramesh Krishnamagaru, Center for Air Sea Technology, Mississippi State University.

Arun Sridharan
B.S. Program, School of Physics
Brandeis University

Title: Computer Applications to Geophysical Sciences

Objective: To explore different areas of computer science and understand how computer technology can be used in geophysical sciences, and to assist CAST in investigating new softwares.

Approach: Current research in oceanography requires the extensive use of computers. Any software that is used to analyze oceanographic data must be able to access information from a database, have a user friendly interface, and be able to display the data in a graphical format using an appropriate model. To understand the software, I studied graphical user interfaces, relational databases, and networking.

Initially, I learned the C programming language and many of the intricacies of the UNIX operating system, and wrote several example programs. Next, I learned X programming and how to use the MOTIF widget set to create widgets serving different functions. To gain more than a theoretical understanding of X programming, I also wrote sample programs that involved several conveniences as well as Xt functions. This included experimentation with the text, pushbutton, bulletin board, dialog, and form widgets. Then, in addition to writing programs with graphical user interfaces in C, I learned how to use Empress, a relational database system. To better understand how information in tables within Empress can be accessed by an application written in C, I wrote programs to open and close tables, retrieve data and stored them in different variables, manipulate the data, and write data back into the tables.

One goal was to write an application that retrieved information from a database, manipulated the data, and allowed the user to choose various operations to be performed on the data in a user friendly manner. Here, I created pushbutton widgets within a window that allowed the user to add, delete, or view information in the database. Next, I opened the database, retrieved the information and stored it in a doubly linked list of structures. When the user clicked one of the options, program control would be transferred to a function that either added, deleted, or displayed the database. This utility was used to extract ocean model data from the database and to evaluate a new graphics utility called ARC INFO. ARC INFO is a geographical information system employed as a standard graphics package at the Naval Oceanographic Office. Another effort was to develop the filters to convert the database to ARC INFO format.

Results: The program developed is user friendly in that all the options are mouse driven. A user does not need to know how a relational database works, and can easily add, delete, and view information in a database over a network without knowing how the different C functions used correlate with one another. Finally, AML based ARC INFO scripts were developed to do grid displays.

Future Research: Could involve translating the C code into C++ and using a different database such as Oracle.

Project Advisor: Mr. Ramesh Krishnamagaru, Center for Air Sea Technology, Mississippi State University.

Clifton Abbott
B.S. Program, MSU Cooperative Education Program
Mississippi State University

Project #1 Title: CAST Model Evaluation System (CMES) Functions

Objectives: To add import, export, and volume functions to the CMES.

Approach: The NEONS BROWER already had a volume function running, and this function was transferred and tailored to fit the CMES.

The import and export functions for CMES were provided in four different file formats. The first format provided was ASCII, which was divided into two formats, text and binary. These formats were decided by the extension of the filename, .dat or .bin. Another format provided was the HDF format. The last format to be completed at a later data is the NetCDF format.

Results: The various functions were added, tested, and integrated into the CMES.

****

Project #2 Title: CAST Model Evaluation System (CMES) Formats

Objectives: To transfer CMES from Kernghan and Ritchie (k&r) format to the ANSI format.

Approach: The main difference between k&r and ANSI was in the function prototyping. All functions had to be prototyped with parameters fully declared within their types. With ANSI's strong type checking, several parameters had to be type casted when being passed to a function or being used in a structure. Along with the ANSI format, CMES was transported to a SGI platform to make it more portable.

Results: CMES, with all functions prototyped, was compiled on a SGI as ANSI code. CMES was also moved back to the standard Unix platform and compiled as ANSI code.

Future Research: CMES in its ANSI format will eventually reside on the ISIS database instead of the NEONS database.

Research Advisor: Mr. Ramesh Krishnamagaru, Center for Air Sea Technology, Mississippi State University.

DISTRIBUTION LIST

1. Scientific Officer (Code 1242)
   Office of Naval Research
   800 N. Quincy Street
   Arlington, VA 22217-5000
   -Mr. Robert Peloquin (3 copies)
   -Dr. Alan Weinstein

2. Defense Technical Information Center
   Building 5, Cameron Station
   Alexandria, VA 22304-6145
   -Two copies

3. Department of the Navy
   Office of Naval Research Resident
   Representative
   101 Marietta Tower-Suite 2805
   101 Marietta Street
   Atlanta, GA 30303

4. Director
   Naval Research Laboratory
   Attention: Code 2627
   Washington, DC 20375

5. Oceanographer of the Navy
   U.S. Naval Observatory
   34th and Massachusetts
   Washington, DC 20392

6. Naval Research Laboratory
   Code 7320
   Stennis Space Center, MS 39529

7. Technical Director
   Naval Oceanographic Office
   Stennis Space Center, MS 39529

8. Commanding Officer
   Fleet Numerical Oceanography Center
   Monterey, CA 93943-5000

9. Technical Director
   Naval Oceanography Command
   Building 1020
   Stennis Space Center, MS 39529

10. CDR David Markham
    Space and Naval Warfare Systems
    Command (PMW165)
    5 Crystal Park, Room 301
    Arlington, VA 22202

11. Manager
    University of Southern Mississippi
    Center for Ocean/Atmospheric Modeling
    Stennis Space Center, MS 39529

12. Engineering Research Center
    Mississippi State University
    Mississippi State, MS 39762
    -Dr. Joe Thompson
    -Dr. Robert Moorhead
    -Mr. Zhifan Zhu

13. Department of Computer Science
    Mississippi State University
    Mississippi State, MS 39762
    -Dr. Don Dearholt
    -Dr. Julia Hodges
    -Dr. Susan Bridges
    -Mr. Chandrashekar Ramanathan
    -Ms. Dongmei Wu
    -Mr. Sridar Koduri

14. Department of Computer Science
    Tulane University
    New Orleans, LA 70118
    -Dr. Fred Petry
    -Mr. Patrick Perrin

15. Cooperative Education Program
    Mississippi State University
    Mississippi, MS 39762-6046
    -Mr. Clifton Abbott
    -Mr. Owen Lagarde

16. Cooperative Education Program
    University of Southern Mississippi
    Hattiesburg, MS 39406-5014
    -Mr. Michael Baer
    -Mr. Mickey Barton

17. Mr. Arun Sridharan
Brandeis University, MB#319
P.o. Box 9110
Waltham, MA 02254

18. Ms. Hollie Molyneux
University of Southern Mississippi
c/o 9810 Ala Moana Place
Diamondhead, MS 39525

19. Dr. Louise Perkins
University of Southern Mississippi
Stennis Space Center, MS 39529

20. Center for Air Sea Technology
Mississippi State University
Stennis Space Center, MS 39529-6000
-Mr. Jim Corbin
-Dr. Lanny Yeske
-Mr. Steve Foster
-Mr. Ramesh Krishnamagaru
-Mr. Valentine Anantharaj
-Mr. Alberto Mestas-Nunez

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. Agency Use Only *(Leave blank)*.	2. Report Date.	3. Report Type and Dates Covered.
	20 SEPTEMBER 1994	TECHNICAL NOTE

4. Title and Subtitle.
1993 STUDENT RESEARCH PROJECTS

5. Funding Numbers.
Program Element No.
Project No.
Task No.
Accession No.

6. Author(s). P. PERRIN, F. PETRY, R. MOORHEAD, Z. ZHU, J. HODGES, M. BAER, M. BARTON, C. RAMANATHAN, D. WU, S. BRIDGES, S. KODURI, V. ANANTHARAJ, R. KRISHNAMAGARU, O. LAGARDE, H. MOLYNEUX, A. SRIDHARAN, S. FOSTER, C. ABBOTT, L. PERKINS, AND L. YESKE (EDITOR)

7. Performing Organization Name(s) and Address(es).
MISSISSIPPI STATE UNIVERSITY
CENTER FOR AIR SEA TECHNOLOGY
STENNIS SPACE CENTER, MS 39529-6000

8. Performing Organization Report Number.
CAST TECHNICAL NOTE 04-94

9. Sponsoring/Monitoring Agency Name(s) and Address(es).
OFFICE OF NAVAL RESEARCH
800 NORTH QUINCY STREET
CODE 1242
ARLINGTON, VA 22217

10. Sponsoring/Monitoring Agency Report Number.
CAST TECHNICAL NOTE 04-94

11. Supplementary Notes.
RESEARCH PERFORMED UNDER THE OFFICE OF NAVAL RESEARCH CONTRACT/GRANT NUMBER NOOO14-92-J-4109

12a. Distribution/Availability Statement.
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED

12b. Distribution Code.

13. Abstract *(Maximum 200 words).*

Eleven 1994 student research projects were sponsored by the Mississippi State University Center for Air Sea Technology. This technical note describes these projects which include an intelligent support system for scientific databases, error pattern identification, scientific visualization of environmental data, object-oriented prototype database, test and evaluation of the CAST Model Evaluation System, enhancement of the Naval Interactive Data Analysis System, remote user enhancements to NEONS, using Oracle to implement a geophysical database, and enhancements to the CMES.

14. Subject Terms.
(U) TECHNICAL NOTE (U) STUDENT (U) RESEARCH (U) CAST

15. Number of Pages. 45

16. Price Code.

17. Security Classification of Report.	18. Security Classification of This Page.	19. Security Classification of Abstract.	20. Limitation of Abstract.
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	